

I found that working with python to write a script that streams data from a file to the MySQL database was easier to do than using PHP. In order to do this I will need to set up python on my RaspberryPi. I used python version 2.7 because it was compatible with the MySQL database that I had installed.

```
pacman -S python2
```

I will also need python2 setups tools to be able to run python2.

```
pacman -S python2-setuptools
```

Next I wanted to create a lightweight development environment to test my python script so I decided to install python-flask. This is an easy to use and create website and lightweight because what I am doing does not require much loadbalancing which is what NginX could be used for.

```
Pacman -S python-flask
```

Now that I have installed my web development environment I am going to test out python and try to read a file into the MySQL database. Two important headers that are needed:

```
from flask import Flask
import mysqlpython
```

And to set the directory path on your website add this line to the beginning of your main:

```
@app.route('/<enter path name>')
```

Also at the end of your code you should include this:

```
If __name__ == '__main__':
    app.run(host='0.0.0.0')
```

```
root@kenny:srv/http
from flask import Flask
import mysqlpython
app = Flask(__name__)

@app.route('/hi')
def asdf():
    msg = converttime()
    tpl = open("template.tpl", "r").read() % msg
    return tpl

def converttime():
    tabledata = "<center><table border = '3'><tr><th>Date</th><th>Time</th><th>Temperature</th></tr>"
    dbdata = mysqlpython.getdata()
    for row in dbdata:
        date = row[0].strftime("%A: %B %d, %Y")
        time = row[0].strftime("%I:%M:%S%p")
        temp = row[1]
        tabledata += "<tr><td>%s</td><td>%s</td><td>%s</td></tr>" % (date, time, temp)
    tabledata += "</table></center>"

    return tabledata

if __name__ == '__main__':
    app.run(host='0.0.0.0')
```

Testpython.py

```
root@kenny:srv/http
import time
import MySQLdb

def mysqlconnect():
    db = MySQLdb.connect(host = "localhost",
                        user = "root",
                        passwd = "root",
                        db = "testDB")

    cur = db.cursor()
    return db, cur

def insertdata():
    db, cur = mysqlconnect()
    datain = [ x for x in open("data.txt", "r").read().split("\n") if x.isdigit() ]
    for i in datain:
        execstring = "INSERT INTO timetemp(time, temperature) VALUES (NOW(), %s);" % i
        cur.execute(execstring)
        time.sleep(2)
    db.commit()
    db.close()

def getdata():
    db, cur = mysqlconnect()
    cur.execute("SELECT * FROM timetemp;")
    db.close()
    return cur.fetchall()
```

Mysqlpython.py

As you can see, I created two .py files.

In the mysqlpython.py file, I have 3 functions. The first function **def mysqlconnect()** allows me to connect to the MySQL database and create a cursor I can use to call MySQL commands. The next function **def insertdata()** allows me to read the text file 'data.txt' and split each data insertion with every break line. I created a for loop to go through the data file and insert each piece of data into the MySQL database as well as the timestamp for when it was inserted. My last function **def getdata()**, retrieves data from the MySQL database by connecting with the database then executing the command **SELECT \* FROM <table name >**.

In the testpython.py file, I used that to insert my python script into an html template file that I use. In the **def asdf()** function I called the function **converttime()** which adjusts the time format from the MySQL table and puts it in an html table I created. I was able to input the data into MySQL database and stream data from the database table to my website.

To start my website up I run the command:

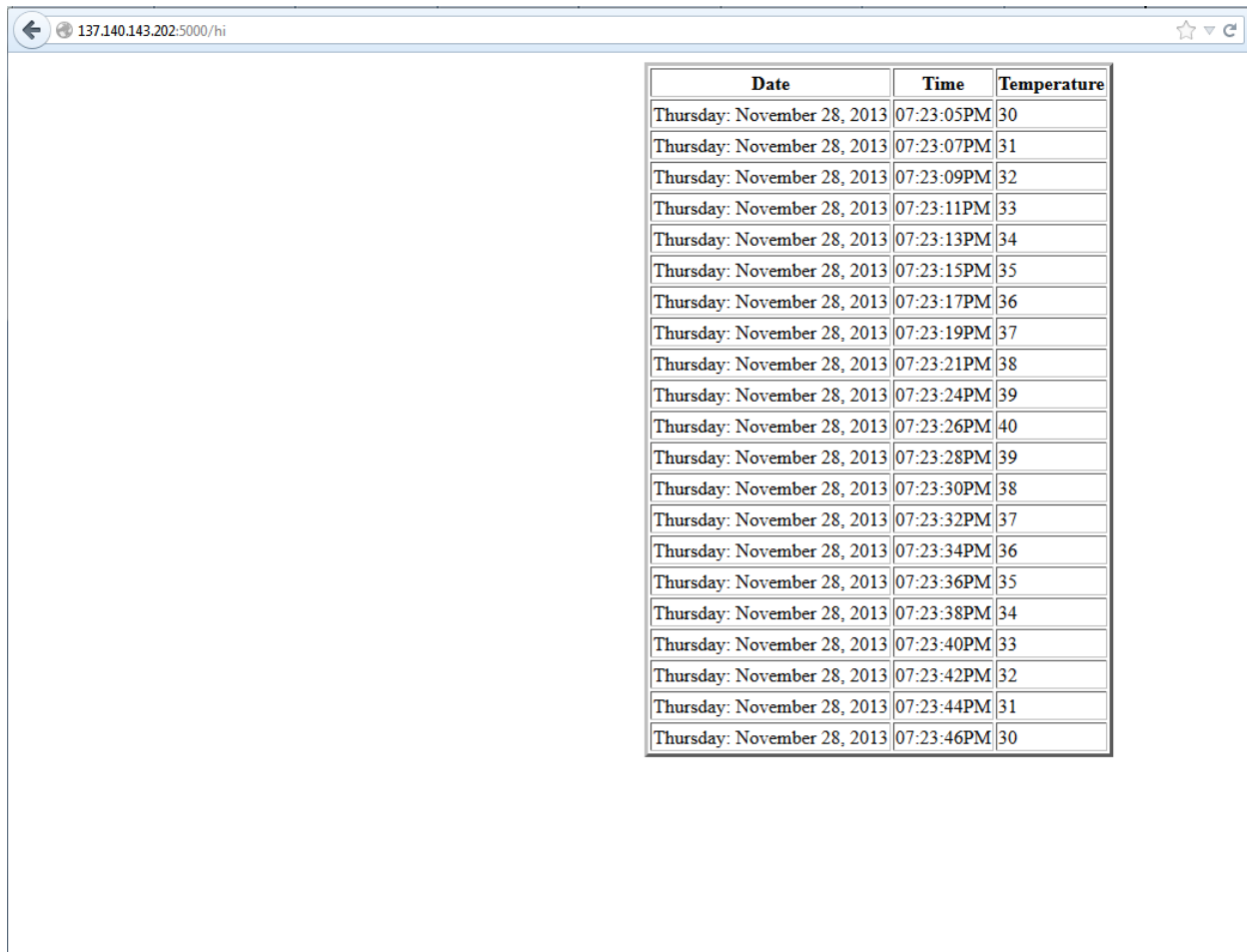
```
python2 testpython.py
```

In my web browser I would enter this to access my website:

```
<ip address>:5000
```

By default flask runs by listening on port 5000.

This is what ends up being displayed on my website.

A screenshot of a web browser window. The address bar shows the URL '137.140.143.202:5000/hi'. The main content area displays a table with three columns: 'Date', 'Time', and 'Temperature'. The table contains 20 rows of data, all for the date 'Thursday: November 28, 2013', with times ranging from 07:23:05PM to 07:23:46PM and temperatures ranging from 30 to 40.

Date	Time	Temperature
Thursday: November 28, 2013	07:23:05PM	30
Thursday: November 28, 2013	07:23:07PM	31
Thursday: November 28, 2013	07:23:09PM	32
Thursday: November 28, 2013	07:23:11PM	33
Thursday: November 28, 2013	07:23:13PM	34
Thursday: November 28, 2013	07:23:15PM	35
Thursday: November 28, 2013	07:23:17PM	36
Thursday: November 28, 2013	07:23:19PM	37
Thursday: November 28, 2013	07:23:21PM	38
Thursday: November 28, 2013	07:23:24PM	39
Thursday: November 28, 2013	07:23:26PM	40
Thursday: November 28, 2013	07:23:28PM	39
Thursday: November 28, 2013	07:23:30PM	38
Thursday: November 28, 2013	07:23:32PM	37
Thursday: November 28, 2013	07:23:34PM	36
Thursday: November 28, 2013	07:23:36PM	35
Thursday: November 28, 2013	07:23:38PM	34
Thursday: November 28, 2013	07:23:40PM	33
Thursday: November 28, 2013	07:23:42PM	32
Thursday: November 28, 2013	07:23:44PM	31
Thursday: November 28, 2013	07:23:46PM	30

One of the best parts about flask is it tells the ip address and the time accessed.

```
[root@kenny http]# python2 testpython.py
* Running on http://0.0.0.0:5000/
137.140.137.118 - - [12/Dec/2013 11:56:14] "GET / HTTP/1.1" 404 -
137.140.137.118 - - [12/Dec/2013 11:56:14] "GET /favicon.ico HTTP/1.1" 404 -
137.140.137.118 - - [12/Dec/2013 11:56:14] "GET /favicon.ico HTTP/1.1" 404 -
137.140.137.118 - - [12/Dec/2013 11:56:18] "GET /hi HTTP/1.1" 200 -
```

Flask is used in most testing environments because you can simply open it up and close it up as well. You can create multiple ones to run on and it is easy to use. NginX is also a great webserver because it allows for load balancing and redirecting traffic to webpages.